(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 168 141 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
02.01.2002 Bulletin 2002/01

(51) Int Cl.[7]: **G06F 1/00**

(21) Application number: **01305484.6**

(22) Date of filing: **25.06.2001**

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE TR
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: **23.06.2000 US 213495 P**

(71) Applicant: **Franklin Electronic Publishers, Incorporated**
Burlington, NJ 08016 (US)

(72) Inventors:
- **Yianilos, Peter**
  Princeton, New Jersey 08540 (US)
- **Kilian, Joseph**
  Princeton Junction, NJ 08550 (US)

(74) Representative: **Hale, Peter et al**
Kilburn & Strode 20 Red Lion Street
London WC1R 4PJ (GB)

(54) A secure and open computer platform

(57) A computer platform is described that provides control features to allow for the protection of intellectual property rights and prevent malfunctioning of the platform. The platform uses 1) a secure operating system including a secure memory management system, 2) public key encryption, 3) data authentication through digital signatures and 4) application/data approval through a flexible access policy through the use of object handlers and an application program approval process. Through these four control features, the platform provides the ability to control access to data and minimize the effects of computer malfunctions.

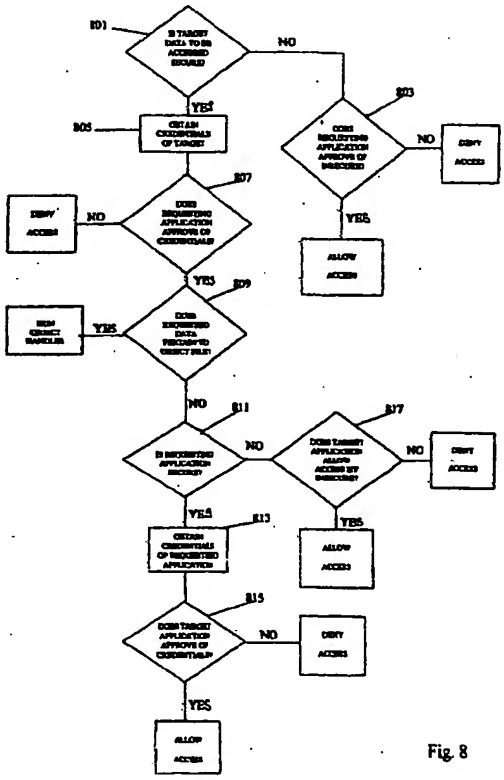

Fig. 8

EP 1 168 141 A2

Description

[0001] This invention relates to an apparatus and system for providing a computer platform, in particular, one that controls the operation of the application programs and object files to adequately protect against computer malfunctions.

[0002] In creating a computer platform, one concern is the amount of control that the platform retains over how application programs operate and data files are accessed on the platform. A computer platform can have control features that implement rules and restrictions on how application programs run and data files are accessed on that platform. Each application program operating on this platform would then usually be written in accordance with these rules or restrictions.

[0003] Control features of a platform are intended to effect some type of security to the platform user and/or the application program or object file writer. For example, a computer platform can contain certain control features to prevent undesirable computer malfunctions such as ones caused by a computer virus or a badly written application program. In other instances, a computer platform can also have control features to prevent violations of a person's intellectual property rights that can occur by unauthorized duplication of copyrighted material.

[0004] The more control features that are implemented in a given platform, the less flexible the platform is in accommodating application programs. If an application program is written to comply with a particular platform's rules, that application program might only be capable of being used on that platform. Conversely, application programs written to comply with a different platform's rules might not operate on this platform.

[0005] An application programmer will usually have to determine which platform the program application is being written for before writing the application program. Consequently, an application programmer will usually favor the platforms that are the least restrictive because it increases the chance of the application program being able to be run on the most platforms. Thus, it is more advantageous for a platform to use the least amount of rules in implementing the desired control features.

[0006] If a computer platform is intended to handle sensitive copyrighted material, then prevention of unauthorized copying becomes paramount and other control issues are less important. With the advent and popularity of digital publications and electronic distribution of publications to be read on electronic readers, protection of copyrights on a computer platform has become important.

[0007] Consequently, it would be advantageous to provide a computer platform that provides control features to prevent unwanted violations of intellectual property rights and still allows enough flexibility for application programs and object files. It would also be advantageous to provide a computer platform that provides control features that protect against malfunctions of program applications on the platform.

[0008] The current invention involves a system that provides enough control features to create a secure platform and yet maintain the flexibility to be able to operate and run a large variety of applications.

[0009] The present invention is defined in the accompanying independent claims. Some preferred features are recited in the dependent claims.

[0010] In a preferred embodiment the invented system entails utilizing control features which in combination protect against malfunctions in a computer platform and provides the ability to prevent unauthorized access to copyrighted material. These security measures include:

1) A secure operating system, including a secure memory management system;
2) Public key encryption;
3) Data authentication through digital signatures; and
4) Application program/object file approval.

[0011] Providing a secure operating system can entail two aspects: 1) ensuring that the operating system is approved for the platform and 2) creating firewalls around application programs and object files that operate on the platform. The term "firewall" is used herein to refer to the arrangement in the computer platform that employs certain memory management components of the operating system to bar access to an application program or an object file in memory. Access to the object file or application program can be granted only if the required permission is obtained.

[0012] Encryption generally entails reformatting data using an "encryption" key such that no one else will be able to utilize or read that data if they do not have an appropriate "decryption" key.

[0013] Data authentication entails the ability of verifying the author and title of data to ensure that the data is properly authored and has not been tampered with from the time of creation by that author to the receipt by the platform.

[0014] Application/data approval allows application programs to reject the type of object files that it will operate on and also allows the object files to reject the application through a flexible access policy through the use of object handlers and an application program approval process.

[0015] In the embodiment of the invented system, these four types of control features are available to be accessed by an application program or data file while on the platform. Application programs and files can operate on the platform without utilizing the control features. Through these four types of control features, a computer platform that is both open to accept various application programs and to protect intellectual property is provided.

[0016] The present invention can be put into practice in various ways, some of which will now be described with reference to the accompanying drawings, in which:

Fig. 1 depicts the layout of an apparatus that implements one embodiment of the present invention;
Fig.2 depicts a flow chart demonstrating the creation of a digital signature in accordance with one embodiment of the present invention;
Fig. 3 depicts a flow chart demonstrating the signature authentication process in accordance with one embodiment of the present invention;
Fig. 4 depicts a flow chart demonstrating the creation of a digital signature along with the further step of encryption in accordance with one embodiment of the present invention;
Fig. 5 depicts a flow chart demonstrating the signature authentication process along with the further step of decryption in accordance with one embodiment of the present invention;
Fig. 6 depicts a flow chart demonstrating the creation of a digital signature along with the further step of encryption in accordance with another embodiment of the present invention
Fig. 7 depicts a flow chart demonstrating the signature authorization process along with the further step of decryption in accordance with another embodiment of the present invention; and
Fig. 8 depicts a flow chart demonstrating the steps of obtaining permission to access data in memory in accordance with one embodiment of the present invention.

[0017] Fig. 1 depicts the layout of a device that implements one embodiment of the present invention. Computer platform 101 is provided with an input interface 103 and an output interface 105 to allow the platform to obtain and transmit data. Input interface 103 and output interface 105 can be the same physical interface so long as it has the capability of both receiving and transmitting data.

[0018] Hardware 107 is contained within computer platform 101 and implements firmware 109. Hardware 107 also comprises memory registers 111. The computer platform's operating system will be loaded onto the hardware 107.

[0019] The operating system ("O/S") is the foundation for the platform's security and digital rights management infrastructure. Firmware 109 contains an O/S verifier 113 to authenticate the O/S before the O/S is loaded onto the hardware 107. Every time that the O/S is loaded or modified, the O/S verifier 113 must authenticate the O/S. O/S verification prevents potential circumvention of the security features that are implemented by the O/S by unauthorized modification or substitution of the O/S. O/S verification does not impose any restrictions that would affect application programs running on the computer platform.

[0020] O/S verification can be accomplished by various methods. One method of O/S verification that can be implemented uses digital signatures. Algorithms for creating and verifying signatures, and for generating the public/private signing key pairs, and well known in the literature (c.f. Bruce, Schneier, "Applied Cryptography, second edition", John Wiley & Sons, Inc. New York (1996). Figs. 2 and 3 depict digital signature verification for the O/S. This technique uses public/secret signature keys. The first step 201 of the digital signature process is to create the data packet to be signed. The data packet is not necessarily just the O/S. Other information may be included as part of the data packet. For example, the credentials of the data can be included along with the data packet. These credentials can include various items to identify the O/S, such as, the author, version and date of creation.

[0021] After determining the data packet to be signed, the next step 203 is to apply a hash function to the data packet. A hash function essentially creates a value of fixed length called the hash value. The hash value is derived from characteristics of the data packet. Different data packets will usually create different hash values. Furthermore, it is computationally unfeasible to produce two different packets that have the same hash value.

[0022] Each author is associated with one or more public/private pairs of signing keys. After the hash value is created (203), the hash value and the private signing key are input to a fixed, publicly known signing algorithm, that produces a digital signature as its output (205). Each private signing key is unique to and known by a single author. The resulting signature is unique to the author who knows the private signing key.

[0023] Fig. 3 depicts the steps that are taken to authenticate a signed data packet. The first step (301) is to recalculate the hash value of the data packet. This calculation reapplies the hash function to the data packet. The recalculated hash value, the signature and the public signing key are given as inputs to a fixed, publicly known signature verification algorithm (303). This algorithm outputs one of two values: "accept", meaning that the signature is to be accepted and "reject", meaning that the signature is to be rejected.

[0024] The combination of creating a hash code and signing the hash code to create the digital signature allows the ability of ensuring that the signer created the signature and that the data packet has not been altered since the signature was created.

[0025] For the O/S verifier, the authorized O/S programmer's public signature key is burned into the hardware of the computer platform. This is necessary since the O/S is usually the first software that is loaded onto the computer platform and the computer platform must be able to verify that O/S when it is loaded. The authorized O/S programmer is usually the manufacturer of the computer platform.

[0026] The procedures of creating a digital signature and authenticating the digital signature detailed in Figs.

2 and 3 are not limited to O/S verification. These procedures can also be applicable to the process of authenticating application programs and object files. However, in addition to the procedures in Figs 2 and 3, authenticating application programs and object files can entail additional steps.

[0027] Figs. 4 and 5 depict the procedures that can be implemented when authenticating signatures of application programs and object files. Similar to the procedures outlined in Fig. 2, the first three steps of creating a signature for application programs and object files are the steps of creating the data packet 401, creating the hash value 403 and creating the digital signature 405. For application programs and object files, credentials should be included with the data packet and those credentials should include at least the name of the author of the data. After the signature of the data packet is created, the data packet and the signature are both encrypted with a public encryption key in step 407.

[0028] Similar to the public/secret signature keys, the public encryption key is distinctively related to a private decryption key. However, both encryption/decryption keys are unique to a particular computer platform rather than a particular programmer. Another difference is that the public encryption key is used to encrypt the data while the private decryption key is used to decrypt the data. The private decryption key is kept secret and is only known to that particular computer platform. This permits only that computer platform access to the encrypted data that has been encrypted with the public key unique to the platform.

[0029] To authenticate the application program or object file, the first step 501 is to decrypt the encrypted data with the computer platform's private decryption key. After the data has been decrypted, the decrypted data should consist of the data packet and the digital signature. The next steps are to recreate the hash value of the data packet 503, and obtain the output from the publicly known signature verification algorithm 505. If the hash values do not match, then the data is erased from memory.

[0030] Encryption of the data packet by using the public/private key technique provides rigorous protection against unauthorized access to the encrypted data by using two separate keys. This process, however, can utilize a substantial amount of the platform's processing resources. In addition, by encrypting the entire data packet, the encryption and decryption process can take a substantial amount of time to perform.

[0031] Another alternative to using the public/private keys to encrypt the data packet is to utilize a less complicated encryption technique in addition to the public/private key encryption. Figs. 6 and 7 depict the process of authenticating program applications and object files utilizing the second encryption technique.

[0032] The first step 601 is to create the data packet. The second and third steps 603, 605 are to create the hash value and digital signatures of the data packet. The fourth step 607 is to encrypt the data packet and signature with a single encryption/decryption key. This single key is used to both encrypt and decrypt. The single decryption key, itself, is then encrypted in the next step 609 with the public encryption key that pertains to the particular platform.

[0033] To decrypt the data at the platform, the platform will first have to decrypt the single encryption/decryption key by using its private decryption key in step 701. With the decrypted single encryption/decryption key, the platform now has the single encryption/decryption key and the data packet can then be decrypted. By separately encrypting the single encryption/decryption key with the public/private keys, the benefit of using the more secure public/private encryption system to protect the entire data packet is gained while maintaining a lower level of complexity.

[0034] Once an application program or object file has been authenticated by the procedures detailed in Figs. 4 or 6 or other similar procedures, it is considered to be secure data. If an application program or object file is not authenticated, it is considered to be insecure.

[0035] By having the encryption/decryption keys unique to each computer platform, the invented system can restrict the availability of data to a particular computer platform. Any computer platform that does not have the correct private decryption key for an encrypted data packet cannot properly decrypt it. This ability, in combination with the signature authentication procedures, provides the ability to exert complete control over data transmissions to the computer platform by: 1) ensuring that any data transmission to the computer platform can only be read by that platform, and 2) ensuring that the data transmission has not been tampered with since the author of the data signed it.

[0036] The computer platform's private decryption key must be burned into the hardware so that the platform will always be able to decrypt encrypted data packets. In addition, authentication of application programs and object file should be done every time that an application program or object files is placed into the computer platform's memory. By performing this authentication every instance that data is loaded onto the platform, the system ensures that all data loaded on the computer platform will be properly characterized as either secure or insecure.

[0037] Although only the manufacturer's public signature is burned into the hardware, the ability to designate application programs and object files as being secure is not restricted to the manufacturer of the computer platform. Any programmer that writes an application program or object file to be used on the computer platform can designate it as being secure data. To accommodate different programmers in designating data as secure, the programmers can send all data to be designated as secure to the manufacturer. After authenticating the data from the programmer, the manufacturer will then digitally sign the data with its secret signature key that cor-

responds to the public signature key that has been burned into the computer platform's hardware.

[0038] Another embodiment entails sending the application programmer's public signature key to the manufacturer. The manufacturer can then digitally sign this public signature key. This signature can then be appended to signatures using the application programmer's signature key. Since it has been signed by the manufacturer, the computer platform will accept the programmer's public signature key as an additional signature key that can be used to authenticate application programs and object files as being secure. These additional public signature keys and the corresponding identities are stored in a list by the O/S. This list will be accessed by the O/S and the identity contained in the credentials will be used to determine the appropriate public signature key.

[0039] This procedure of approving another programmer's public signature key is not applicable to O/S verification. It is important to always control the O/S because it ensures that the security features imposed by the O/S cannot be circumvented.

[0040] One security feature implemented by the O/S that should not be circumvented is creating firewalls around data in the memory registers 111 to preserve the integrity and privacy of the application programs and the object files. Both an application program's memory and an object files' memory are automatically shielded from all other applications that may be running on the computer platform 101. No special programming by the programmers is needed to enjoy this protection. To breach a firewall, the O/S will seek permission from two places: 1) the application program requesting the breach and 2) the data in memory that is to be accessed.

[0041] Fig. 8 depicts the process of obtaining the appropriate permission before allowing access to application programs or object files in memory. For the first step 801, the O/S determines if the data in memory to be accessed belongs to a secure object file or application program. If the data in memory is not secure, then the O/S informs the requesting application program that it is not secure. The O/S will permit the requesting application program to gain access to the insecure data if the requesting application has been written to allow access to insecure data as determined in step 803.

[0042] If the data in memory pertains to secure data, then the O/S provides the credentials of the secure target data to the requesting application program in step 805 in order for the requesting application program to determine if the credentials are approved in step 807.

[0043] The O/S will not automatically provide access to the secure data if the requesting application program approves of the credentials. With secure data, the O/S then determines whether the data pertains to an application program or an object file in step 809.

[0044] If the memory belongs to an object file, then the O/S will usually run an object handler associated with the object file. The term "object handler" used herein refers to a program that is associated with object files that determine if access to that object file is permitted. For example, an object handler associated with a particular object file might permit access through the fire wall for an application program which derives from the publisher of the object file. The object handler can use a number of parameters (publisher, expiration date, platform identifications, etc.) as criteria for access. If the object handler determines that permission to access the desired memory is permitted, the O/S will allow access.

[0045] Object handlers are usually created by the author of the data packet and are included along with the data packet that will be digitally signed. However, default object handlers can created by the O/S for secure object files that have no object handler in another embodiment of the present invention.

[0046] If the target data in memory belongs to an application program, then the O/S must then determine if the target application approves of the requesting application program. The next step 811 is to determine if the requesting application is secure. Depending on the requesting application's secure status, either the credentials of the requesting application is communicated to the target application program in step 813 or the fact that it is insecure is communicated to the target application program. In either case, the target program application must determine if access to its memory is granted as depicted in steps 815 and 817 respectively.

[0047] Obtaining permission to breach a firewall must be obtained every new instance in which an application program requests access to data in memory. Once an application program obtains permission to access the memory, the application program does not have to obtain permission again until it terminates its access. Once access is terminated, a successive request for access to data in memory will treated as a new instance and the requisite permission must be obtained.

[0048] By rigorously protecting the integrity of application programs and object files in memory, the system minimizes the damage caused by computer malfunctions. Any malfunctioning application program or computer viruses can be blocked from affecting other program applications and object files. The firewall essentially isolates potential adverse effects to application programs and object files.

[0049] Firewalls also allow the system to extend the control over data transmissions exerted through data encryption and signature authentication to cover the data while in the computer platform's memory. Once in the computer platform's memory, unauthorized access is blocked by the combination of the firewalls and the approval process through object handlers and application program approvals. Thus, unauthorized access to the data is prevented from the time of the creation of the digital signature through to the transmission of the data to the computer platform and use on the platform.

[0050] Unauthorized access to the data can also be prevented when the data is exported out of the computer

platform by applying the encryption/decryption process to all data being transmitted out of the computer platform. Every time that any data marked as secure is transferred out of the computer platform, the computer platform encrypts the data with the computer platform's encryption key. By encrypting the data, this prevents anyone who does not have that computer platform's decryption key from improperly accessing the data that has been transferred out of the platform.

[0051] A variety of data may be designated as sensitive by the OS, meaning that it is undesirable for other entities to learn the contents of this data. For example, the OS may designate as sensitive data that it receives in encrypted form. The OS may automatically deem portions of the memory used by a secure application to be sensitive. Or it may act on a request, generated by an application or the OS itself, that data be designated as being sensitive.

[0052] Normally, sensitive date is kept within the computer platform. However, due to memory limitations or to enable backups, it may be necessary to export sensitive data from the computer platform. In the preferred embodiment of the invention, information that is designated as sensitive by the OS is encrypted when it is transmitted, in its entirety or in part, out of the computer platform, and decrypted when it is reimported into the computer platform. In the preferred embodiment of the invention, the OS performs these encryption and decryption steps automatically. Other modes of operation are possible and may be more suitable for specific contexts. For example, the OS may query the computer platform's user or applications residing on the computer platform before performing the encryption or decryption operations. By controlling access to data that has been transferred out of the platform, the invented system effects control over access to the data in all instances of potential access after the digital signature is created. Besides fraudulently securing permission by accessing the secret and private keys, a person must pass the implemented security measures to gain access to secure data. These security features do not impose restrictive rules upon the application programmer because insecure application programs and object files can still operate on the invented system.

[0053] The present invention is not to be considered limited in scope by the preferred embodiments described in the specification. The software for the system by which the method of the invention is executed can be stored on any suitable computer readable medium such as floppy disc, computer hard drive, CD-ROM, Flash ROM, non-volatile ROM and RAM. The medium can be magnetically or optically readable. Additional advantages and modifications, which readily occur to those skilled in the art from consideration and specification and practice of this invention are intended to be within the scope of the following claims.

Claims

1. A data control system comprising:

   a computer platform operable to authenticate an operating system to be loaded on the platform and preventing the operating system from being loaded onto the platform when the operating system is not authenticated;
   a memory in which application programs and object files can be stored, the operating system being operable to create a firewall around data in memory pertaining to application programs and object files to control access to the application programs and object files;
   an input interface connected with the platform to allow input data to be received by the platform, the operating system being capable of decrypting the input data and of authenticating the input data, and the firewalls around the data in memory being capable of allowing the application programs to access the data in memory when approval of access is obtained from the application program and from the data in memory; and
   an output interface connected to the platform to allow the platform to transmit output data out of the platform, which output data is encrypted when transmitted.

2. A data control system as claimed in claim 1, wherein the platform authenticates the operating system by verifying a digital signature associated with the operating system.

3. A data control system as claimed in claim 1 or 2, wherein the operating system decrypts the input data with a private decryption key unique to the platform.

4. A data control system as claimed in claim 3, further comprising a sending station capable of encrypting data with a public encryption key; said public encryption key being directly related to said private decryption key of said computer platform.

5. A data control system as claimed in any of claims 1 to 4, wherein the operating system authenticates the input data by verifying a digital signature associated with the input data with a public signature key and input data that is not authenticated by said operating system is classified as insecure data.

6. A data control system as claimed in any of claims 1 to 5, wherein the output interface encrypts the output data when the output data includes at least a portion of data that has been authenticated by the operating system.

7. A data control system as claimed in claim 5, further comprising a sending station capable of creating a digital signature with a secret signature key; the secret signature key being distinctively associated with the sending station.

8. A data control system as claimed in any of claims 1 to 4, wherein the operating system is capable of authenticating the input data by using a hash function.

9. A data control system as claimed in any of claims 1 to 8, wherein the data in memory gives approval for access through an object handler associated with each of the object files when the data in memory pertains to the object files.

10. A data control system as claimed in any of claims 1 to 9, wherein the output data is encrypted with a public encryption key unique to the platform.

11. A data control system as claimed in claim 10, wherein the output data is decrypted with a private decryption key associated with the public encryption key.

12. A data control system as claimed in any of claims 1 to 11 in which the computer platform includes hardware for authenticating the operating system and preventing the operating system being loaded when it is not authenticated.

13. A data control system comprising:

a sending station, including: (a) a plurality of application programs, (b) a plurality of object files, (c) a plurality of handler programs, each associated with a separate one of said object files; and (d) a plurality of secret key encoded signatures, each distinctive to a subset of said application programs and said object files;
a plurality of receiving platforms, each having firmware and an operating system, the firmware being operable to authenticate the operating system,
each of the receiving platforms being adapted to receive the application programs, object files, handlers and signatures, and each of the receiving platforms having: (a) a public signature identification key to authenticate the signatures and (b) firewalls associated with the application programs and object files to control access to each of the application programs and object files, one of the handler programs associated with each of the object files being operable to permit access to the associated object files by an appropriate one or more of the application programs; and each of the handler programs being programmable to permit multi-parameter control over access to the associated object files.

14. The data control system of claim 13 wherein: the object files and the application programs at the sending station are encrypted with a public key unique to the receiving platform being addressed and wherein the encrypted object files and application programs are decrypted with a private key, at the receiving platform.

15. The data control system of claim 13 or 16, wherein the signature identification is provided through a signature creation algorithm and a secret key at the sending station, and through a signature verification algorithm and a public key at each receiving platform.

16. The system of any of claims 13 to 15 wherein:

the sending station has a plurality of secret key encoded signatures, each signature being distinctive to a separate set of application programs and data texts,
each receiving platform having a plurality of public signature identification keys to correspond to the plurality of secret keys at the sending station.

17. A method for providing a data control system, comprising the steps of:

authenticating an operating system to be loaded on a computer platform; the authenticating step to be performed every time an operating system is loaded on the computer platform;
verifying credentials of data transmitted to the computer platform before loading the data into memory of the computerplatform;
creating firewalls around data loaded into memory of the computer platform;
decrypting data transmitted to the computer platform with a private decryption key unique to the computer platform; and
encrypting data transferred out of the computer platform with a public encryption key unique to the computer platform and associated with the public decryption key.

18. A method as claimed in claim 17, wherein the authenticating step is performed by verifying a digital signature associated with the operating system.

19. A method as claimed in claim 17 or 18, further comprising the step of obtaining permission before allowing an application program to access data loaded into memory.

20. A method as claimed in claim 19 wherein the obtaining step is performed through object handlers.

21. A computer readable medium having computer-executable instructions for performing the method of any of claims 16 to 20.

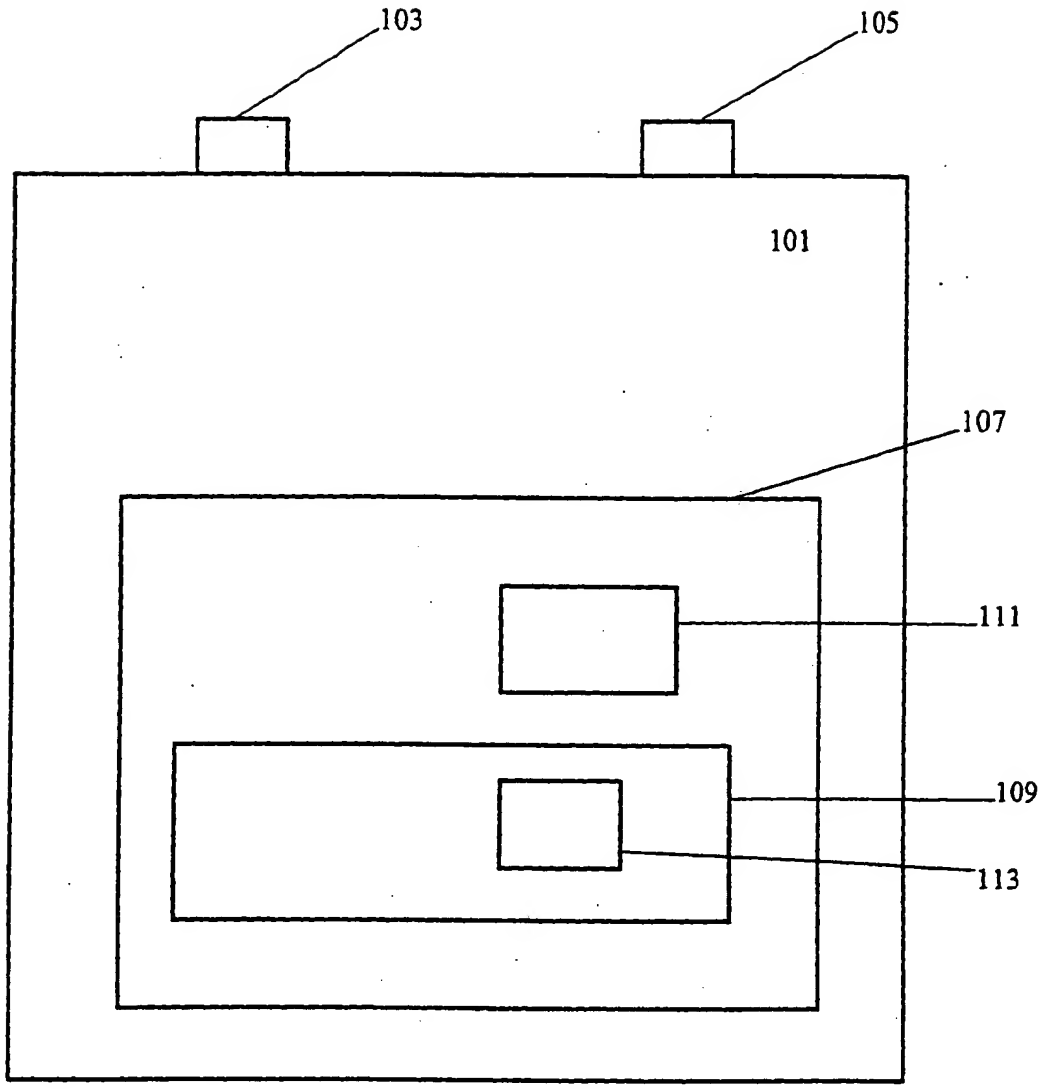

Fig 1

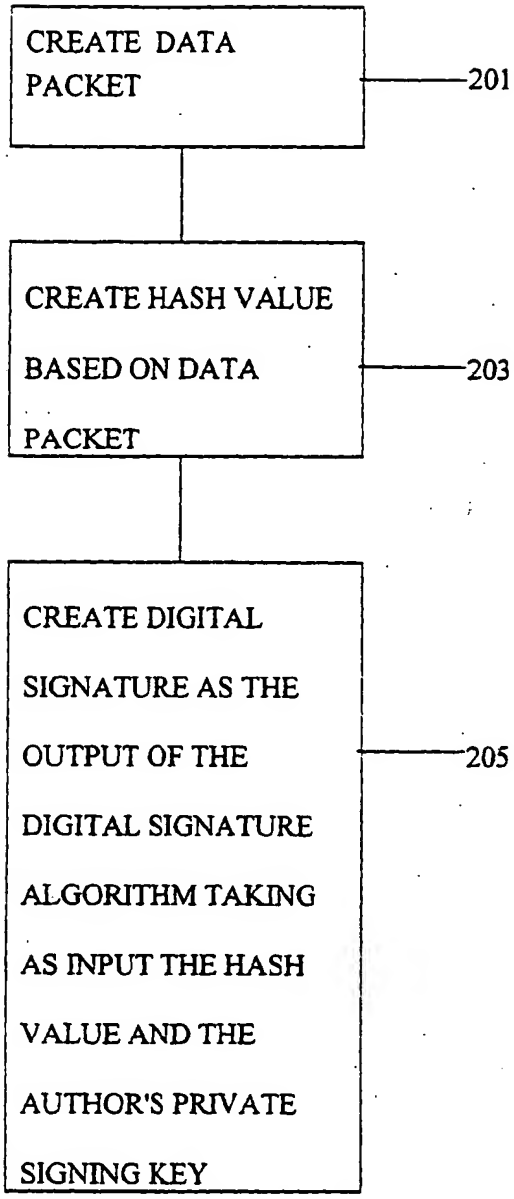
CREATE DATA PACKET
201
CREATE HASH VALUE BASED ON DATA PACKET
203
CREATE DIGITAL SIGNATURE AS THE OUTPUT OF THE DIGITAL SIGNATURE ALGORITHM TAKING AS INPUT THE HASH VALUE AND THE AUTHOR'S PRIVATE SIGNING KEY
205

Fig. 2

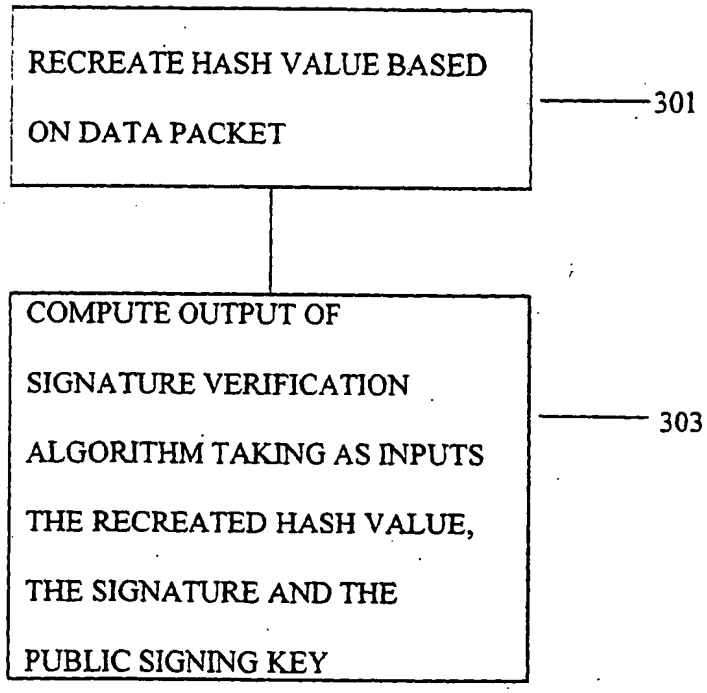

Fig. 3

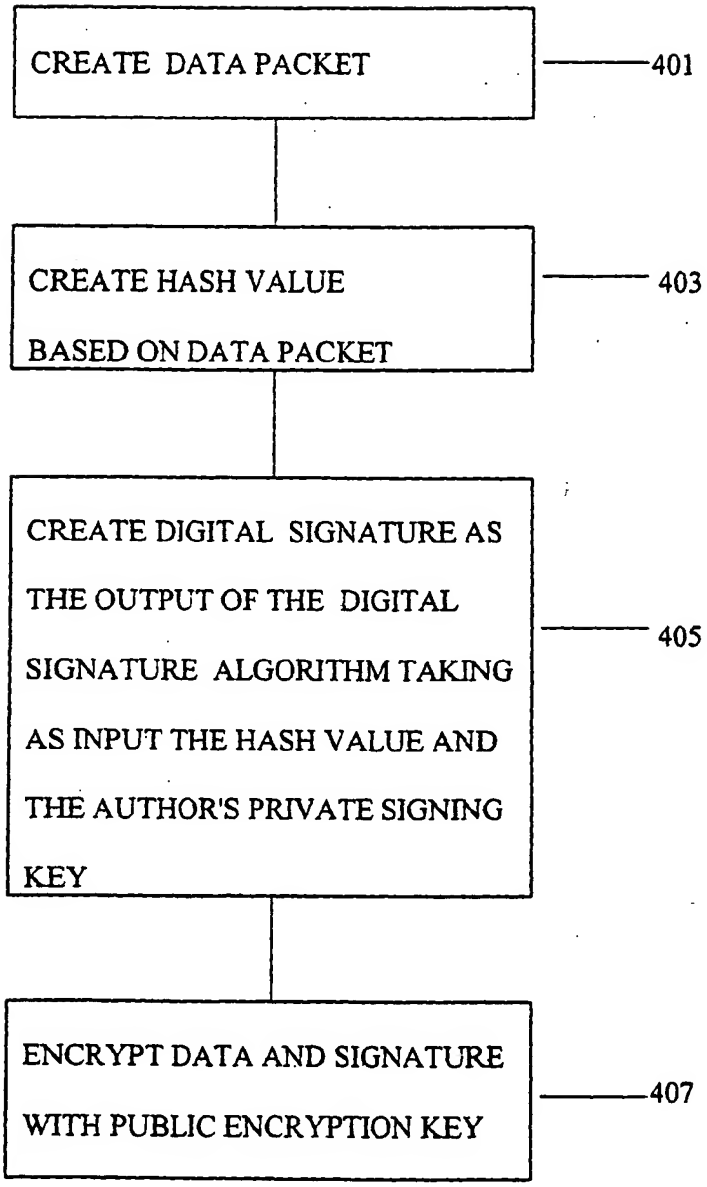
CREATE DATA PACKET
401
CREATE HASH VALUE BASED ON DATA PACKET
403
CREATE DIGITAL SIGNATURE AS THE OUTPUT OF THE DIGITAL SIGNATURE ALGORITHM TAKING AS INPUT THE HASH VALUE AND THE AUTHOR'S PRIVATE SIGNING KEY
405
ENCRYPT DATA AND SIGNATURE WITH PUBLIC ENCRYPTION KEY
407

Fig. 4

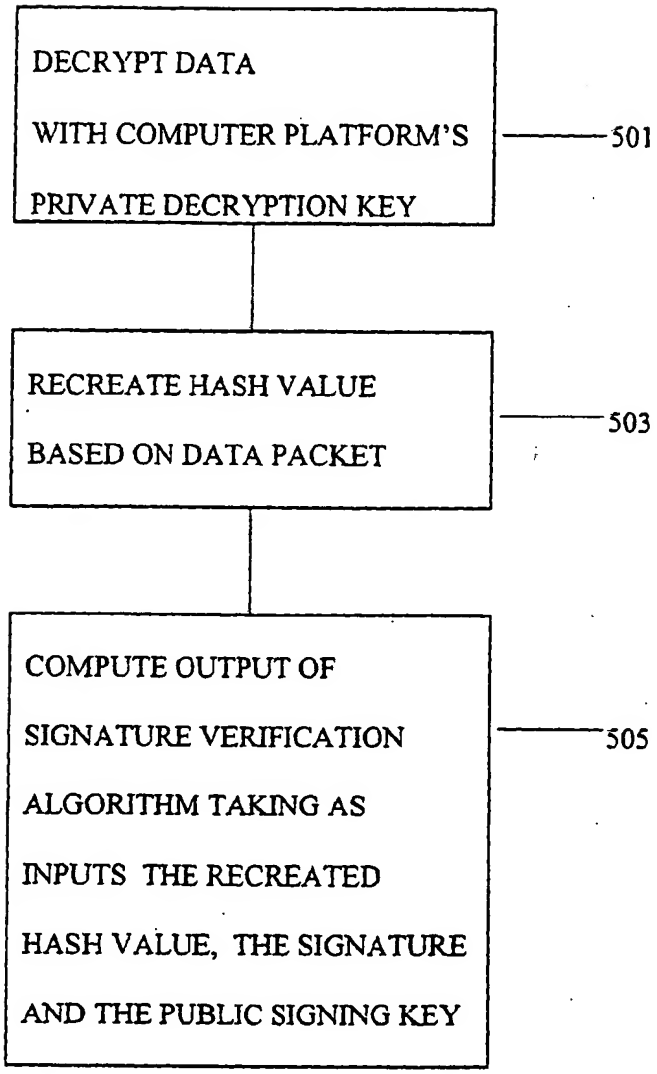

Fig. 5

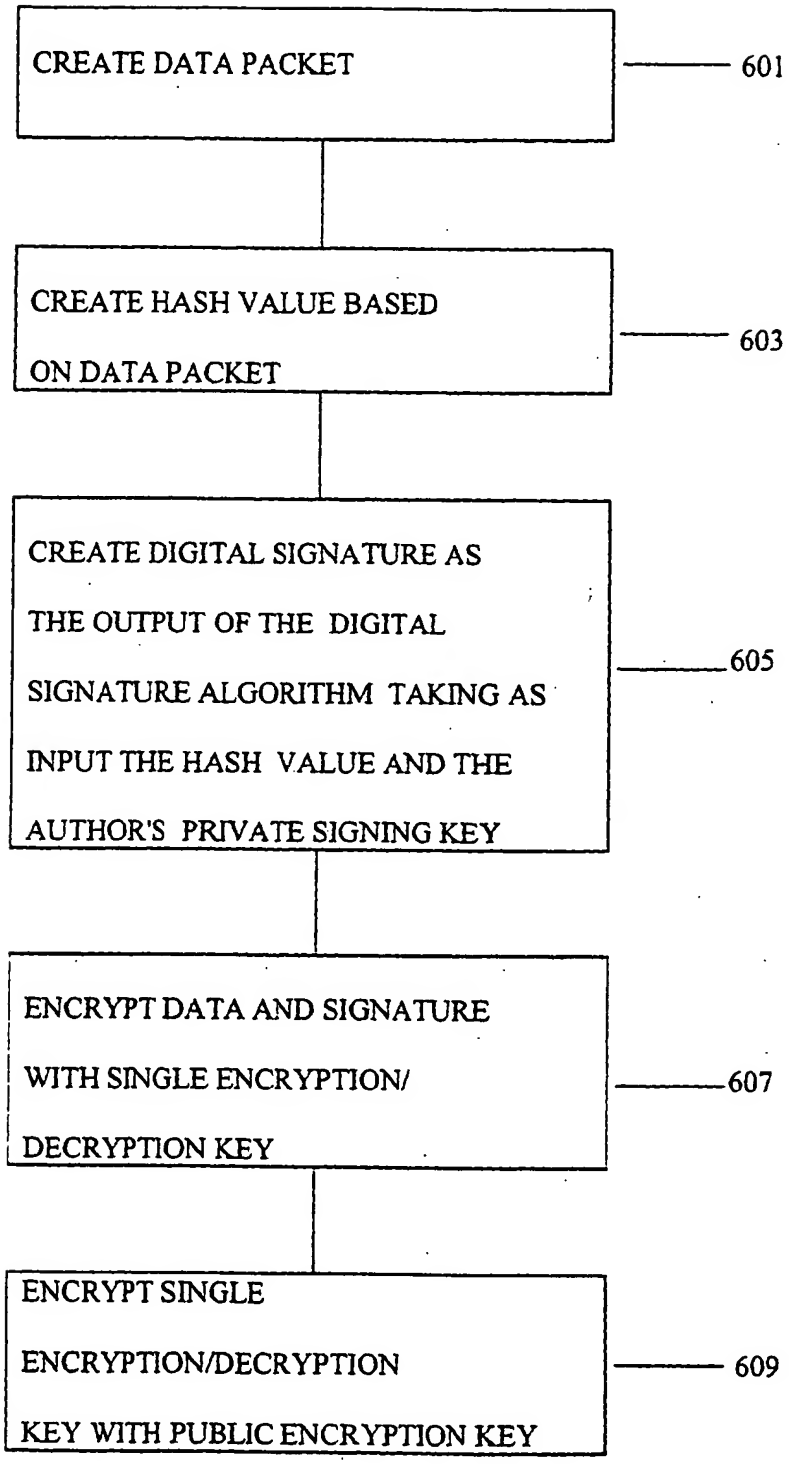
CREATE DATA PACKET
601
CREATE HASH VALUE BASED ON DATA PACKET
603
CREATE DIGITAL SIGNATURE AS THE OUTPUT OF THE DIGITAL SIGNATURE ALGORITHM TAKING AS INPUT THE HASH VALUE AND THE AUTHOR'S PRIVATE SIGNING KEY
605
ENCRYPT DATA AND SIGNATURE WITH SINGLE ENCRYPTION/ DECRYPTION KEY
607
ENCRYPT SINGLE ENCRYPTION/DECRYPTION KEY WITH PUBLIC ENCRYPTION KEY
609

Fig. 6

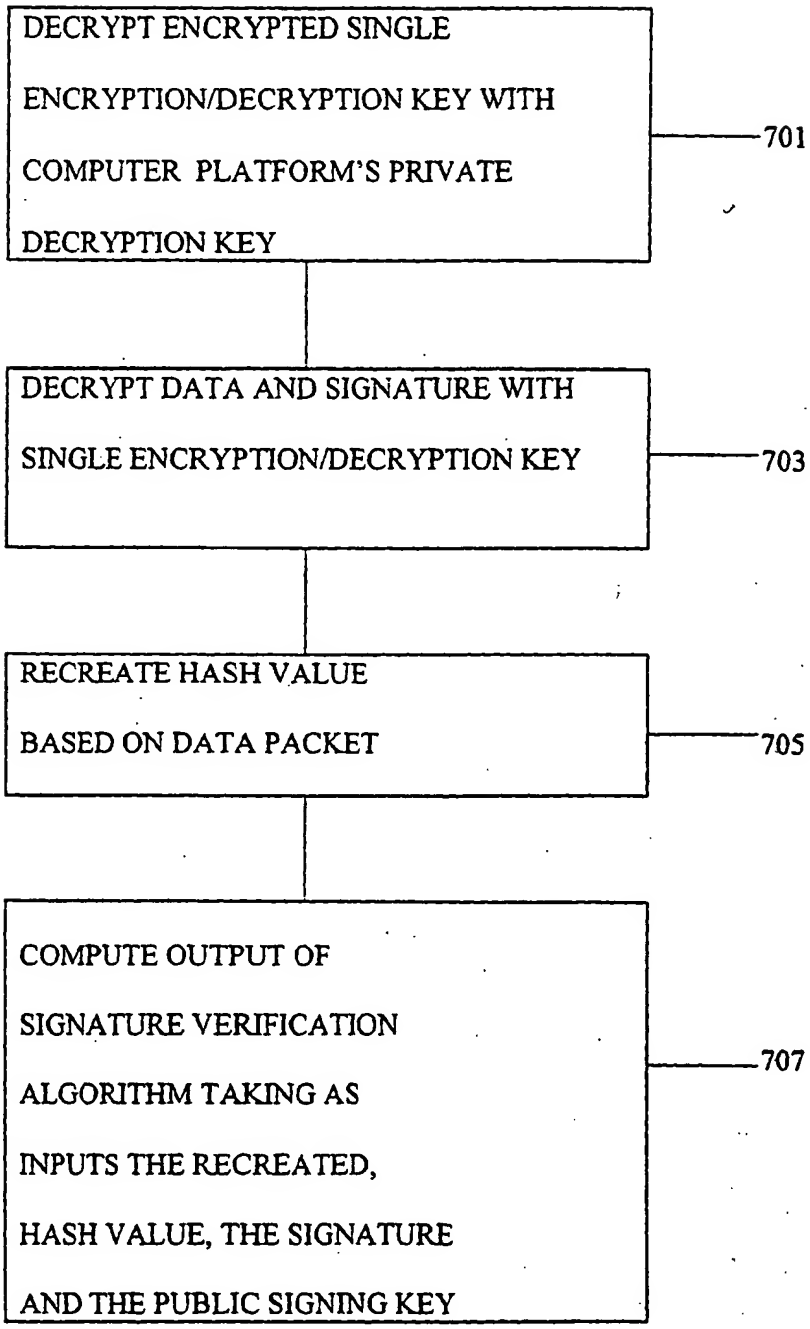

Fig. 7

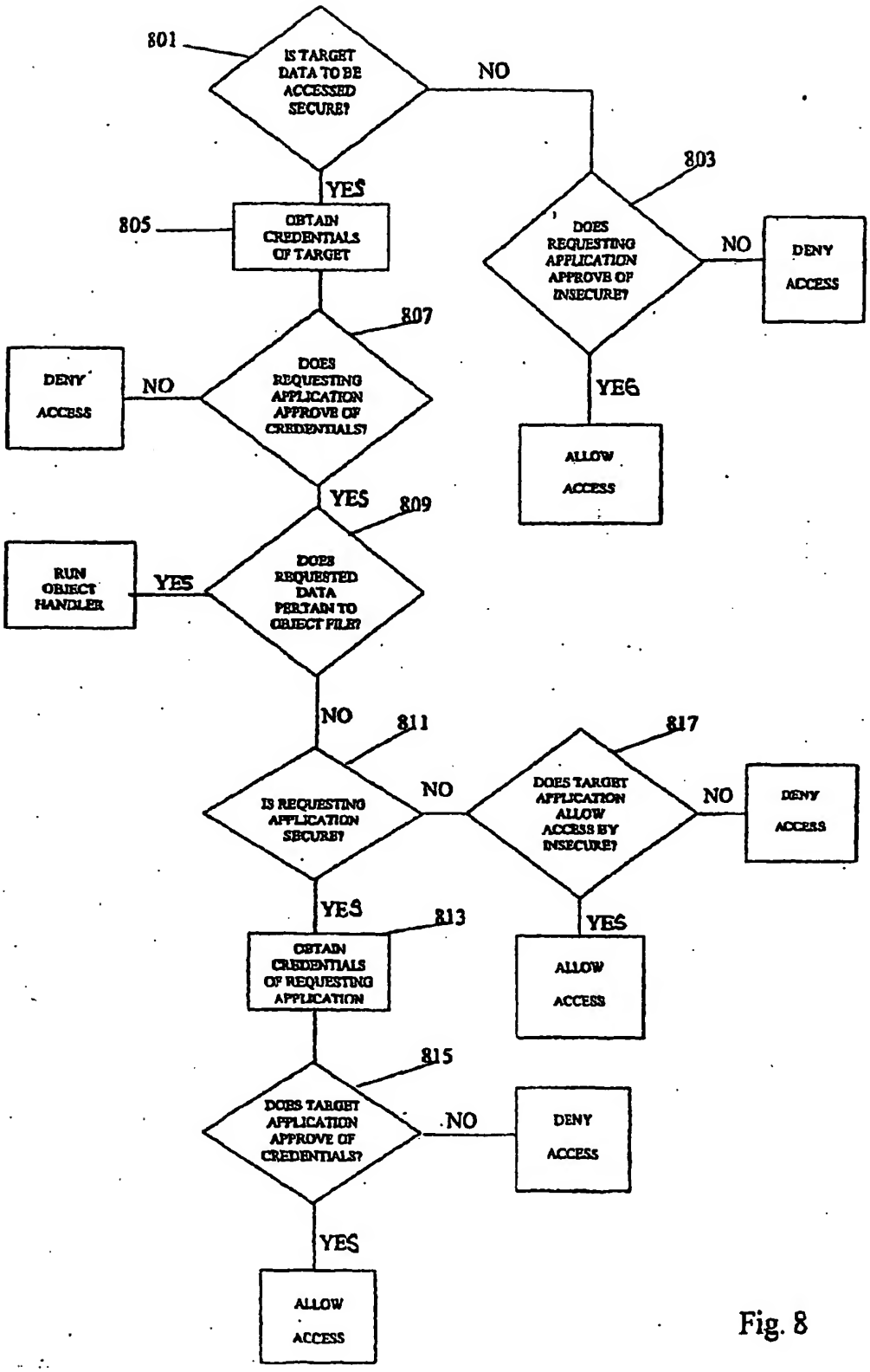
801
IS TARGET DATA TO BE ACCESSED SECURE?
NO
YES
803
DOES REQUESTING APPLICATION APPROVE OF INSECURE?
NO
DENY ACCESS
YES
ALLOW ACCESS
805
OBTAIN CREDENTIALS OF TARGET
807
DOES REQUESTING APPLICATION APPROVE OF CREDENTIALS?
NO
DENY ACCESS
YES
809
DOES REQUESTED DATA PERTAIN TO OBJECT FILE?
YES
RUN OBJECT HANDLER
NO
811
IS REQUESTING APPLICATION SECURE?
NO
817
DOES TARGET APPLICATION ALLOW ACCESS BY INSECURE?
NO
DENY ACCESS
YES
ALLOW ACCESS
YES
813
OBTAIN CREDENTIALS OF REQUESTING APPLICATION
815
DOES TARGET APPLICATION APPROVE OF CREDENTIALS?
NO
DENY ACCESS
YES
ALLOW ACCESS

Fig. 8

(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 168 141 A3

(12) **EUROPEAN PATENT APPLICATION**

(88) Date of publication A3:
08.09.2004 Bulletin 2004/37

(43) Date of publication A2:
02.01.2002 Bulletin 2002/01

(21) Application number: **01305484.6**

(22) Date of filing: **25.06.2001**

(51) Int Cl.7: **G06F 1/00**

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE TR
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: **23.06.2000 US 213495 P**

(71) Applicant: **Franklin Electronic Publishers, Incorporated**
Burlington, NJ 08016 (US)

(72) Inventors:
- **Yianilos, Peter**
  Princeton, New Jersey 08540 (US)
- **Kilian, Joseph**
  Princeton Junction, NJ 08550 (US)

(74) Representative: **Hale, Peter et al**
Kilburn & Strode
20 Red Lion Street
London WC1R 4PJ (GB)

(54) A secure and open computer platform

(57) A computer platform is described that provides control features to allow for the protection of intellectual property rights and prevent malfunctioning of the platform. The platform uses 1) a secure operating system including a secure memory management system, 2) public key encryption, 3) data authentication through digital signatures and 4) application/data approval through a flexible access policy through the use of object handlers and an application program approval process. Through these four control features, the platform provides the ability to control access to data and minimize the effects of computer malfunctions.

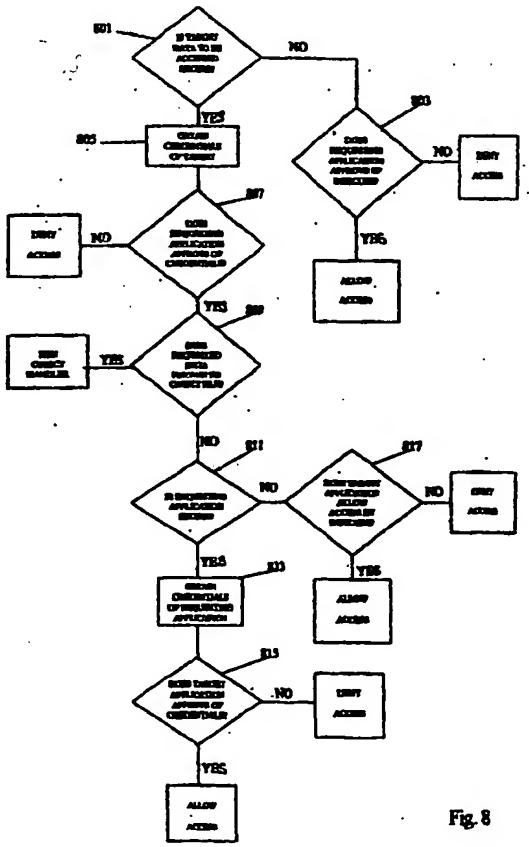

Fig. 8

EP 1 168 141 A3

European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 01 30 5484

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.7)
Y	WO 98/45768 A (NORTHERN TELECOM LTD) 15 October 1998 (1998-10-15) * abstract * * figures 1-3 * -----	2-4,7, 10,12, 14,18	G06F1/00 G06F21/00 G06F12/14
X	WO 00/10283 A (HANNAH ERIC C ; INTEL CORP (US)) 24 February 2000 (2000-02-24) * abstract * * page 3, paragraph 3 - page 4, paragraph 1 * * page 5, line 3 - line 14 * * page 3, paragraph 2 * * page 2, line 27 - page 3, line 1 * -----	1,5,6,8, 9,11,13, 15-17, 19-23	
Y		2-4,7, 10,12, 14,18	

TECHNICAL FIELDS SEARCHED (Int.Cl.7)

G06F

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
Munich	15 July 2004	Kleiber, M

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

ANNEX TO THE EUROPEAN SEARCH REPORT ON EUROPEAN PATENT APPLICATION NO. EP 01 30 5484

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report. The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

15-07-2004

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 9845768 A	15-10-1998	US 6108420 A	22-08-2000
		AU 6492198 A	30-10-1998
		CA 2285392 A1	15-10-1998
		WO 9845768 A1	15-10-1998
		CN 1255209 T	31-05-2000
		EP 0974084 A1	26-01-2000
		JP 2002503365 T	29-01-2002
WO 0010283 A	24-02-2000	US 6735696 B1	11-05-2004
		AU 5479799 A	06-03-2000
		WO 0010283 A1	24-02-2000

EPO FORM P0459

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82